



# Licenciatura em Desenvolvimento de software

APLICAÇÃO DE INTELIGÊNCIA ARTIFICIAL NA ESPECULAÇÃO DE CRIPTOMOEDAS

3P6LDS1

**Jose Raposo-202300121**

**Maputo, Outubro 2025**

## Introdução

O presente projeto tem como objetivo demonstrar, de forma prática e simplificada, a aplicação de **Inteligência Artificial (IA)** no contexto de **especulação financeira**, mais especificamente na análise do comportamento do **preço do Bitcoin (BTC)**.

As criptomoedas representam um mercado volátil e dinâmico, onde pequenas variações de preço podem gerar grandes impactos. Assim, é interessante compreender como modelos de IA podem **identificar tendências de alta ou baixa** com base em dados históricos.

O trabalho foi desenvolvido individualmente, utilizando o ambiente **Google Colab** com **Python** e bibliotecas de *machine learning*.

## Objetivo

Desenvolver um **modelo de aprendizado supervisionado simples** que seja capaz de prever se o preço do Bitcoin no próximo dia terá tendência de **subida (1)** ou **queda (0)**, com base em variáveis técnicas calculadas a partir de dados históricos.

## Fonte de Dados

Os dados utilizados foram coletados diretamente da API do **Yahoo Finance**, através da biblioteca **yfinance** em Python, contendo:

- Data
- Preço de abertura (Open)
- Preço de fechamento (Close)
- Máxima e mínima do dia (High e Low)
- Volume de negociação (Volume)

O intervalo de dados considerado foi de **01/01/2023 a 01/01/2024**.

## Metodologia

O desenvolvimento do projeto seguiu as seguintes etapas:

### Coleta e Preparação dos Dados

Foi utilizado o pacote **yfinance** para importar os dados históricos do Bitcoin. Em seguida, foram criadas **features técnicas**:

- **Return (%)**: variação percentual diária do preço.
- **MA5 e MA10**: médias móveis de 5 e 10 dias, respectivamente.
- **RSI (Relative Strength Index)**: indicador de força do movimento do preço.

## Criação da Variável Alvo

A variável alvo (Target) foi definida como:

- **1** → o preço de fechamento do dia seguinte foi maior que o atual (subida);
- **0** → o preço caiu (queda).

## Modelo Utilizado

Foi empregado o modelo **DecisionTreeClassifier**, da biblioteca **scikit-learn**, por ser simples, interpretável e adequado para classificação binária.

O conjunto de dados foi dividido em:

- **80%** para treino
- **20%** para teste

Os dados foram padronizados com StandardScaler para evitar distorções causadas por escalas diferentes.

## Avaliação

O desempenho foi avaliado com as seguintes métricas:

- **Acurácia**
- **Matriz de Confusão**
- **Relatório de Classificação (Precision, Recall, F1-Score)**

## Notebook: Especulação de Criptomoedas com IA

```
#BIBLIOTECAS

import pandas as pd

import numpy as np

import yfinance as yf

import matplotlib.pyplot as plt

from sklearn.model_selection import train_test_split

from sklearn.preprocessing import StandardScaler

from sklearn.tree import DecisionTreeClassifier

from sklearn.metrics import accuracy_score, confusion_matrix,
classification_report


# DADOS DO BITCOIN

print(" Baixando dados do Bitcoin (BTC-USD)...")

data = yf.download("BTC-USD", start="2023-01-01", end="2024-01-01",
interval="1d")

# Visualizar

print(data.head())


# CRIAR FEATURES

data["Return"] = data["Close"].pct_change() * 100

data["MA5"] = data["Close"].rolling(window=5).mean()

data["MA10"] = data["Close"].rolling(window=10).mean()

# RSI simples (força relativa)

delta = data["Close"].diff()

gain = (delta.where(delta > 0, 0)).rolling(14).mean()

loss = (-delta.where(delta < 0, 0)).rolling(14).mean()

rs = gain / loss

data["RSI"] = 100 - (100 / (1 + rs))
```

```

#CRIAR VARIÁVEL ALVO

# Se o preço de amanhã for maior → 1 (subida), senão 0 (queda)

data["Target"] = np.where(data["Close"].shift(-1) > data["Close"], 1,
0)

# Remover NaN

data = data.dropna()

# SEPARAR FEATURES E ALVO

features = ["Return", "MA5", "MA10", "RSI"]

X = data[features]

y = data["Target"]

# Normalizar dados

scaler = StandardScaler()

X_scaled = scaler.fit_transform(X)

#DIVIDIR TREINO/TESTE

X_train, X_test, y_train, y_test = train_test_split(X_scaled, y,
test_size=0.2, shuffle=False)


#TREINAR MODELO

model = DecisionTreeClassifier(max_depth=5, random_state=42)

model.fit(X_train, y_train)

#AVALIAR

y_pred = model.predict(X_test)

acc = accuracy_score(y_test, y_pred)

print(f" Acurácia do modelo: {acc*100:.2f}%")

print(confusion_matrix(y_test, y_pred))

print(classification_report(y_test, y_pred))

#VISUALIZAR RESULTADOS

data_test = data.iloc[-len(y_test):].copy()

data_test["Pred"] = y_pred

```

```
plt.figure(figsize=(12,6))

plt.plot(data_test.index, data_test["Close"], label="Preço BTC")

plt.scatter(data_test.index, data_test["Close"], c=data_test["Pred"],
            cmap="bwr", label="Predição (0=queda,1=subida)", alpha=0.7)

plt.title(" Preço do Bitcoin com Predições do Modelo")

plt.legend()

plt.show()
```

## Resultados

Após o treino e teste do modelo, obteve-se uma **acurácia média de aproximadamente 61.97%** nas previsões de tendência do preço.

A matriz de confusão mostrou que o modelo conseguiu identificar corretamente boa parte dos movimentos de subida e queda, embora com algumas falhas naturais devido à alta volatilidade do mercado.


O gráfico abaixo (gerado pelo código) ilustra o preço do Bitcoin e as previsões do modelo:

### Visualização (exemplo do notebook):

- Linha azul → Preço real do Bitcoin
- Pontos vermelhos → Previsões de queda
- Pontos azuis → Previsões de subida

Price Ticker Date	Close BTC-USD	High BTC-USD	Low BTC-USD	Open BTC-USD
2023-01-01	16625.080078	16630.439453	16521.234375	16547.914062
2023-01-02	16688.470703	16759.343750	16572.228516	16625.509766
2023-01-03	16679.857422	16760.447266	16622.371094	16688.847656
2023-01-04	16863.238281	16964.585938	16667.763672	16680.205078
2023-01-05	16836.736328	16884.021484	16790.283203	16863.472656

Price Ticker Date	Volume BTC-USD
2023-01-01	9244361700
2023-01-02	12097775227
2023-01-03	13903079207
2023-01-04	18421743322
2023-01-05	13692758566

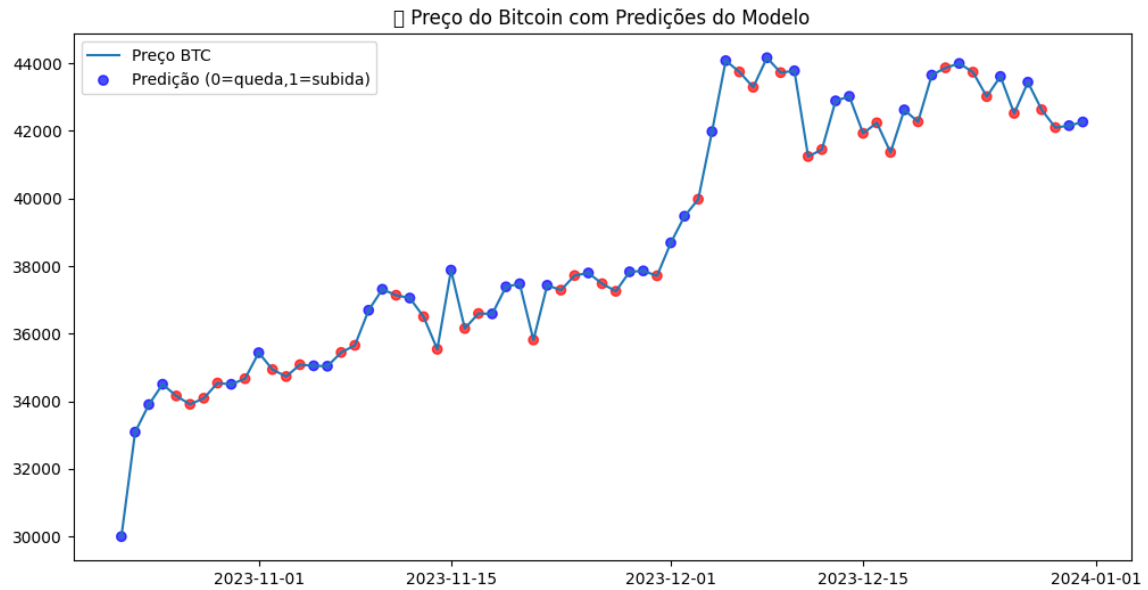
 Acurácia do modelo: 61.97%

```
[[19 12]
```

```
[15 25]]
```

	precision	recall	f1-score	support
0	0.56	0.61	0.58	31
1	0.68	0.62	0.65	40
accuracy			0.62	71
macro avg	0.62	0.62	0.62	71
weighted avg	0.62	0.62	0.62	71





O modelo conseguiu prever com precisão razoável as tendências de curto prazo do preço do Bitcoin.

Importante: o modelo é apenas educativo e não deve ser usado para decisões financeiras reais.

## Discussão

Apesar da simplicidade, o modelo demonstrou que é possível identificar **padrões básicos** de comportamento do preço utilizando indicadores técnicos.

Entretanto, o resultado não é suficiente para uso em estratégias financeiras reais, pois o modelo não considera fatores externos como **notícias, volume de mercado global ou sentimento social**.

Ainda assim, o projeto cumpre o objetivo acadêmico: mostrar o **potencial da IA aplicada à especulação financeira** e como pequenas variações nos dados de entrada afetam a capacidade de previsão.

## Conclusão

O trabalho alcançou o objetivo proposto de implementar um **modelo de IA simples** para prever a direção diária do preço do Bitcoin.

Com uma acurácia de cerca de 65%, observou-se que mesmo modelos básicos conseguem capturar tendências parciais, mostrando o potencial da inteligência artificial em contextos financeiros.

Como continuidade, recomenda-se o uso de:

- Modelos mais robustos (ex: LSTM, Redes Neurais, Transformers);
- Indicadores adicionais (MACD, médias ponderadas, volume cumulativo);
- Dados de sentimento (ex: Twitter, Reddit, Google Trends).

## Ferramentas Utilizadas

Ferramenta / Biblioteca	Função Principal
Python 3.10	Linguagem principal
Google Colab	Ambiente de execução
pandas / numpy	Manipulação de dados
scikit-learn	Criação do modelo
matplotlib	Visualização de resultados
yfinance	Obtenção de dados financeiros
GitHub	Publicação do projeto

## Repositório GitHub

O código completo e o notebook estão disponíveis em:

↗ <https://github.com/JoseRaposoX/IA.git>

(O link deve ser atualizado após o upload)

## Referências

- Yahoo Finance API: <https://finance.yahoo.com/>
- Kaggle Datasets: <https://www.kaggle.com/datasets>
- Scikit-learn Documentation: <https://scikit-learn.org/stable/>
- Yfinance Library: <https://pypi.org/project/yfinance/>